



2183  
520.41205X00

RECEIVED  
MAR 21 2002  
Technology Center 2100

Applicant(s): AKASHI, et al

Serial No.: 10 / 076,547

Filed: FEBRUARY 19, 2002

Title: A PROCESS SCHEDULING METHOD BASED ON ACTIVE  
PROGRAM CHARACTERISTICS ON PROCESS EXECUTION,  
PROGRAMS USING THIS METHOD AND DATA PROCESSORS

LETTER CLAIMING RIGHT OF PRIORITY

Assistant Commissioner for  
Patents  
Washington, D.C. 20231

MARCH 18, 2002

Sir:

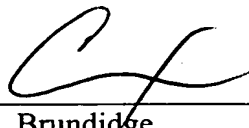
Under the provisions of 35 USC 119 and 37 CFR 1.55, the applicant(s) hereby claim(s)  
the right of priority based on:

Japanese Patent Application No. 2001 - 192174  
Filed: JUNE 26, 2001

A certified copy of said Japanese Patent Application is attached.

Respectfully submitted,

ANTONELLI, TERRY, STOUT & KRAUS, LLP

  
\_\_\_\_\_  
Carl I. Brundidge  
Registration No. 29,621

CIB/rp  
Attachment



NT0606US

日 本 国 特 許 庁  
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出 願 年 月 日  
Date of Application:

2001年 6月26日

出 願 番 号  
Application Number:

特願2001-192174

[ST.10/C]:

[JP2001-192174]

出 願 人  
Applicant(s):

株式会社日立製作所

RECEIVED

MAR 21 2002

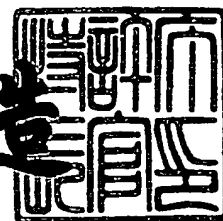
Technology Center 2100

CERTIFIED COPY OF  
PRIORITY DOCUMENT

2002年 3月 1日

特 許 庁 長 官  
Commissioner,  
Japan Patent Office

及 川 耕 造



出証番号 出証特2002-3011663

【書類名】 特許願

【整理番号】 H01003681A

【あて先】 特許庁長官 殿

【国際特許分類】 G05F 15/16

【発明者】

    【住所又は居所】 東京都国分寺市東恋ヶ窪一丁目 2 8 0 番地 株式会社日立製作所中央研究所内

    【氏名】 明石 英也

【発明者】

    【住所又は居所】 東京都国分寺市東恋ヶ窪一丁目 2 8 0 番地 株式会社日立製作所中央研究所内

    【氏名】 上原 敬太郎

【発明者】

    【住所又は居所】 東京都国分寺市東恋ヶ窪一丁目 2 8 0 番地 株式会社日立製作所中央研究所内

    【氏名】 田中 剛

【特許出願人】

    【識別番号】 000005108

    【氏名又は名称】 株式会社 日立製作所

【代理人】

    【識別番号】 100075096

    【弁理士】

    【氏名又は名称】 作田 康夫

    【電話番号】 03-3212-1111

【手数料の表示】

    【予納台帳番号】 013088

    【納付金額】 21,000円

【提出物件の目録】

    【物件名】 明細書 1

【物件名】 図面 1  
【物件名】 要約書 1  
【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 プロセス実行時のプログラム動作特性に基づくプロセススケジューリング方法及びこれを用いたプログラム及びデータ処理装置

【特許請求の範囲】

【請求項 1】

複数のプロセッサを含み、該複数のプロセッサの少なくとも一部にはそれぞれ当該プロセッサのプログラム実行中におけるプロセッサ動作特性を採取する性能測定手段をそれぞれ有する計算機システムにおいて、実行するプロセスを前記複数のプロセッサのいずれかに割り付けるスケジューリング方法であって、

前記プロセッサのいずれかでプロセスを実行する際に前記性能測定手段を制御して、当該プロセスの前記プロセッサ動作特性を採取すること、

前記計算機上で実行中若しくは実行可能な各プロセスの前記プロセッサ動作特性に基づき、各プロセスを割り付けるプロセッサを優先的に選択すること、

との手順を有するプロセススケジューリング方法。

【請求項 2】

前記プロセッサ動作特性として、プログラム実行時間に占めるメモリアクセス待ち時間の比率を使用することを特徴とする請求項 1 記載のプロセススケジューリング方法。

【請求項 3】

前記プロセッサ動作特性として、プログラム実行中におけるメモリアクセス量を使用することを特徴とする請求項 1 記載のプロセススケジューリング方法。

【請求項 4】

前記各プロセスの割り付けでは、実行中もしくは実行可能な各プロセスの前記メモリアクセス待ち時間の比率もしくは前記メモリアクセス量が大きい順に、各プロセスをキャッシュ容量が大きいプロセッサに優先して割り付けることを特徴とする請求項 2 もしくは請求項 3 のいずれかに記載のプロセススケジューリング方法。

【請求項 5】

前記計算機システム上で実行中若しくは実行可能な各プロセスの前記メモリア

クセス待ち時間の比率が大きい順に、各プロセスをメモリアクセスレイテンシが小さいプロセッサに優先して割り付けることを特徴とする請求項 2 記載のプロセススケジューリング方法。

【請求項 6】

前記計算機システムは、1 以上のプロセッサからなるノードを複数有し、

前記各プロセスの割り付けでは、前記計算機システム上で実行中若しくは実行可能な各プロセスの前記メモリアクセス量に基づき、各ノードに割り付ける 1 以上のプロセスのメモリアクセス量合計が、当該ノードのメモリアクセス性能を超えない様に優先して割り付けることを特徴とする請求項 3 記載のプロセススケジューリング方法。

【請求項 7】

前記性能測定手段を制御して採取した各プロセスのプロセッサ動作特性をファイルシステム上に記録する手順を更に有し、

次に当該プロセスを実行する際に、前記ファイルシステムに記録した当該プロセスのプロセッサ動作特性に基づき、当該プロセスを割り付けるプロセッサを優先的に選択することを特徴とする請求項 1 記載のプロセススケジューリング方法。

【請求項 8】

前記性能測定手段を制御して各プロセスのメモリアクセス特性の変化を採取し、  
各プロセス毎に前記プロセッサのタイムスライスを割り当てる際に、前記計算機上で実行中若しくは実行可能な各プロセスの前記メモリアクセス特性の変化に基づき、各プロセスに割り当てるタイムスライスの長さを変更することを特徴とする請求項 2 乃至請求項 3 記載のプロセススケジューリング方法。

【請求項 9】、

タイムスライス内におけるプロセスのメモリアクセス待ち時間の比率又はメモリアクセス量が、あらかじめ指定された若しくは各プロセスのメモリアクセス特性を元にスケジューリング機能が決定したしきい値を超える減少傾向にあることを検出し、当該プロセスのタイムスライスの長さを既定値より大きな値に変更す

ることを特徴とする請求項 8 記載のプロセススケジューリング方法。

【請求項 1 0】

前記性能測定手段を制御して各プロセスのメモリアクセス量の変化を採取し、  
計算機内の各プロセッサに割り付けられる各プロセスについて、タイムスライスの開始時間を異なる時刻に設定し、タイムスライスを同時に開始した場合と比較し同時に動作中のプロセスのメモリアクセス量合計が計算機のメモリアクセス性能を超過することに伴う性能低下を抑えることを特徴とする請求項 3 記載のプロセススケジューリング方法。

【請求項 1 1】

複数のプロセッサを有する計算機システムであって、  
前記プロセッサの各々は、当該プロセッサ内で生じる複数の事象の中から特定の事象の生起回数をカウントする性能測定データレジスタと、前記性能測定レジスタで測定する事象を指示するための、性能測定制御レジスタの組からなる性能測定回路を 1 以上有し、

前記性能測定回路は、前記計算機システムのメモリ上に設けた性能測定用領域に性能測定データレジスタの値を順次格納していくことにより、特定の事象のタイムスライス内での変化を採取可能とした計算機システム。

【請求項 1 2】

前記プロセッサの一部が性能測定手段を持たない場合、性能予測手段を有するプロセッサでプロセスを実行した際に採取したメモリアクセス特性に基づき、各プロセスを割り付けるプロセッサを優先的に選択することを特徴とする請求項 1 記載のプロセススケジューリング方法。

【発明の詳細な説明】

【 0 0 0 1 】

【発明の属する技術分野】

本発明は、複数プロセッサを有する計算機システムにおけるプロセススケジューリング方式に関し、特にプロセス実行時のプロセッサ又はシステム動作特性を動的に採取しこれを基にスケジューリングを行うプロセススケジューリング方法及びこれを用いた計算機システムに関する。

## 【0002】

## 【従来の技術】

近年、ネットワークビジネス市場の急拡大やネットワークコンピューティングの高度化に伴い、計算機システムに要求される性能が急速に増加している。既存の計算機への投資を活かしつつ計算機システムの性能向上を図るには、既存計算機へのプロセッサ等の増設、又は、既存計算機システムへの計算機の追加が有効である。

## 【0003】

一方、半導体デバイスの急速な発展により、プロセッサや計算機自体の性能向上も急ピッチで進んでいる。従って先に述べた計算機システムの性能向上においては、既存の計算機システムのプロセッサ又は計算機と比較し高い性能を持つプロセッサ又は計算機を増設するのが望ましい。

## 【0004】

現在のオペレーティングシステムでは、複数プロセッサを有する並列計算機においてプログラムを実行する際に、プロセススケジューラがプログラムを構成するプロセス、スレッド（又は軽量プロセス）毎にプロセッサに割り付け処理をさせる。又、クラスタソフトウェアでは、複数計算機からなる計算機システム（クラスタシステム）においてプログラムを実行する際に、スケジューラがプロセス、スレッド毎に計算機に割り付け処理をさせる。クラスタソフトウェアの例は、「SUN Microsystems, SUN Cluster Architecture: A White Paper, Cluster Computing, 1999, Proceedings, 1st IEEE Computer Society, International Workshop on, page 331-338」が挙げられる。

## 【0005】

先に述べた様な異なる性能特性を持つプロセッサや計算機が混在するクラスタシステムにおいては、各プロセスを当該プロセスの実行に適したプロセッサに割り付けることができれば、より高性能を発揮できる。特開平11-31134（従来技術1）では、異なる仕様の複数プロセッサを有する計算機において、各プ



プログラムにプログラムの実行特性を示すプログラム属性情報を付加して、スケジューラは各プロセッサの性能特性を示すプロセッサ特性情報及びプログラム属性情報に基づいて各プログラムを最適なプロセッサで実行する方法が示されている。従来技術1では、プログラム属性情報は具体的には画像処理、通信処理、高速の技術計算処理、音声処理、またはマルチメディア情報処理などのデータ処理の形態または処理するデータの種類などを示す一種のフラグ情報であるとしている。

## 【0006】

## 【発明が解決しようとする課題】

従来技術1は、異なる性能特性を持つプロセッサが混在する計算機において、あらかじめ付加したプロセッサ特性情報及びプログラム属性情報を基にプロセススケジューリングを行うものである。このため、異なる性能特性を持つ計算機が混在するクラスタシステムにおいて、動的なプログラム特性に基づき最適なプロセススケジューリングを行うためには以下に示す課題を解決する必要がある。

(1) 従来技術1においては、プログラムに対応するプログラム属性情報をあらかじめ付加しておく必要がある。このため、プログラムを実際に動作させた際に初めて判明する動的なプログラム特性を基にプロセススケジューリングを行えない。

(2) 従来技術1においては、異なる性能特性を持つ計算機が混在するクラスタシステムのプロセススケジューリングを考慮していない。

(3) 計算機の処理性能は、プロセッサ内部の処理性能とプロセッサ外部の処理性能（主にメモリスistem性能）によって決まる。従来技術1においては、プロセッサの特性情報に基づきプロセススケジューリングを行うため、計算機のメモリアクセス特性に応じたプロセススケジューリングを考慮していない。

## 【0007】

本発明は、従来技術で解決されていない上記課題の解決し、異なる性能特性を持つプロセッサが混在する計算機や、異なる性能特性を持つ計算機が混在するクラスタシステムにおける高度なプロセススケジューリング方式を提供する。

## 【0008】

## 【課題を解決するための手段】

本発明で開示する、上記課題を解決するための代表的構成は以下のものが挙げられる。

## 【0009】

計算機システムに含まれる複数のプロセッサの少なくとも一部に、当該プロセッサのプログラム実行中におけるプロセッサ動作特性を採取する性能測定手段を設け、前記プロセッサのいずれかでプロセスを実行する際に前記性能測定手段を制御して、当該プロセスの前記プロセッサ動作特性を採取し、前記計算機上で実行中若しくは実行可能な各プロセスの前記プロセッサ動作特性に基づき、各プロセスを割り付けるプロセッサを優先的に選択する。

前記プロセッサ動作特性としては、例えばプログラム実行時間に占めるメモリアクセス待ち時間の比率、プログラム実行中におけるメモリアクセス量が使用可能である。一例では、前記計算機システム上で実行中若しくは実行可能な各プロセスの前記メモリアクセス待ち時間の比率又は前記メモリアクセス量が大きい順に、各プロセスをキャッシュ容量が大きいプロセッサに優先して割り付ける。また別の例では、前記計算機上で実行中若しくは実行可能な各プロセスの前記メモリアクセス待ち時間の比率が大きい順に、各プロセスをメモリアクセスレイテンシが小さいプロセッサに優先して割り付ける。

## 【0010】

また、前記計算機上で実行中若しくは実行可能な各プロセスの前記メモリアクセス量に基づき、各ノードに割り付ける1以上のプロセスのメモリアクセス量合計が、当該ノードのメモリアクセス性能を超えない様に優先して割り付ける。

## 【0011】

さらに、前記性能測定手段を制御して、各プロセスのメモリアクセス特性の変化を採取することにより、各プロセス毎に前記プロセッサのタイムスライスを割り当てる際に、前記計算機上で実行中若しくは実行可能な各プロセスの前記メモリアクセス特性の変化に基づき、各プロセスに割り当てるタイムスライスの長さを変更する。

## 【0012】

タイムスライス内におけるプロセスのメモリアクセス待ち時間の比率又はメモリアクセス量が、あらかじめ指定された若しくは各プロセスのメモリアクセス特性を元にスケジューリング機能が決定したしきい値を超える減少傾向にあることを検出し、当該プロセスのタイムスライスの長さを既定値より大きな値に変更する。

## 【 0 0 1 3 】

前記性能測定手段を制御して、各プロセスのメモリアクセス量の変化を採取し、計算機システム内の各プロセッサに割り付けられる各プロセスについて、タイムスライスの開始時間を異なる時刻に設定し、タイムスライスを同時に開始した場合と比較し同時に動作中のプロセスのメモリアクセス量合計が計算機のメモリアクセス性能を超過することに伴う性能低下を抑える。

この様な、プロセッサ動作特性の変化に基づくプロセススケジューリングを効率良く実現するために、前記プロセッサは、当該プロセッサ内で生じる複数の事象の中から特定の事象の生起回数をカウントする性能測定データレジスタと、前記性能測定レジスタで測定する事象を指示するための、性能測定制御レジスタの組からなる性能測定回路を1以上有し、前記性能測定回路は、前記計算機のメモリ上に設けた性能測定用領域に性能測定データレジスタの値を順次格納していくことにより、特定の事象のタイムスライス内での変化を採取可能とする性能測定方式を実現する。

そして、前記性能測定手段を制御して採取した各プロセスのプロセッサ動作特性をファイルシステム上に記録し、次に当該プロセスを実行する際に、前記ファイルシステムに記録した当該プロセスのプロセッサ動作特性に基づき、当該プロセスを割り付けるプロセッサを優先的に選択する方法や、前記プロセッサの一部が性能測定手段を持たない場合でも、性能測定手段を有する前記プロセッサでプロセスを実行した際に採取したメモリアクセス特性に基づき、各プロセスを割り付けるプロセッサを優先的に選択可能とする。

## 【 0 0 1 4 】

以上のプロセススケジューリング方法は、単一の計算機に限らず、複数計算機をネットワークにより結合した計算機クラスタシステムに対しても容易に適用で

きる。

【0015】

【発明の実施の形態】

以下、図面を用いて本発明の実施の形態を説明する。

《実施の形態例1》

本発明の第1の実施の形態を図1～12を用いて説明する。

【0016】

図1は、本発明に係る計算機クラスタシステムのハードウェア及びソフトウェア構成要素の関係の概略図を示す。

【0017】

図1の計算機クラスタシステムは、計算機1(110-1)、計算機2(110-2)、…、計算機m(110-m)がネットワーク(100)により接続された構成を採る。各計算機(110-1、…、110-m)上では、各々1つのオペレーティングシステム(160-1、…、160-m)及び複数のプロセス(170-11～170-1L1、170-21～170-2L2、…、170-m1～170-mLm)が動作する。ここでプロセスとは、アプリケーションプログラムをプロセッサに割り当て可能な単位に分割した実行単位である。本発明では、プロセスは一般にスレッド又は軽量プロセスと呼ばれているものを含む、広い意味でのプロセスを指す。

【0018】

各計算機(110-1、…、110-m)は、各々プロセッサ11(120-11)～1N1(120-1N1)、プロセッサ21(120-21)～2N2(120-2N2)、…、プロセッサm1(120-m1)～120-mNm)を有する。各プロセッサ(120-11、…、120-mNm)は、各々性能測定手段(130-11、…、130-mNm)を持ち、メモリアクセス待ち時間、メモリアクセス量等プロセッサ内部で発生する種々の事象を計測できる。このような性能測定手段は、例えばIntel社の「Pentium Proファミリディベロッパーズマニュアル下巻 第10章」に開示されている公知の技術である。

## 【0019】

各計算機（110-1、…、110-m）上で動作するオペレーティングシステム（160、…、1～160-m）は、各プロセス（170-11～170-1L1、…、170-m1～170-mLm）をプロセッサ（120-11～120-1N1、…、120-m1～120-mNm）に割り付けるスケジューリング機能を有する。本実施の形態例では、各プロセス（170-11～170-mLm）は処理開始時点又は処理の途中で任意のプロセッサ（120-11～120-mNm）に移送（マイグレート）して実行可能であることを前提とする。この様なプロセス移送を実現する方法は、前川他編の「分散オペレーティングシステム（共立出版株式会社） 第5章」に開示されている公知の技術である。

## 【0020】

各オペレーティングシステム（160-1～160-m）のスケジューリング機能は、後述する協調動作により計算機間にまたがったプロセス移送を伴う動的負荷分散を行う。本実施の形態例ではこれを実現するため、計算機1（110-1）のオペレーティングシステム1（160-1）にクラスタスケジューラ（150）、各計算機（110-1、…、110-m）のオペレーティングシステム（160-1、…、160-m）にクラスタノードスケジューラ（140-1、…、140-m）を設ける。

## 【0021】

クラスタスケジューラ（150）は、計算機クラスタシステム内で実行される各プロセスをいずれかの計算機（110-1～110-m）に割り当てる機能を持つ。この割り当ての決定においては、従来のプロセススケジューラの一般的なアルゴリズムに加え、プロセス実行時に性能測定手段（130-11～130-mNm）を使用して採取したプロセス（170-11～170-mLm）毎のプロセス動作特性を考慮する。

## 【0022】

クラスタノードスケジューラ（140-1、…、140-m）は、クラスタスケジューラ（150）が対応する計算機（110-1、…、110-m）に割り当てた各プロセスを当該計算機（110-1、…、110-m）内のいずれかの

プロセッサ (120-11~120-1N1、…、120-m1~120-mNm) に割り当てる機能を持つ。この割り当ての決定においては、従来のプロセススケジューラの一般的なアルゴリズムに加え、プロセス実行時に性能測定手段 (130-11~130-mNm) を使用して採取したプロセス (170-11~170-mLm) 毎のプロセス動作特性を考慮する。また、クラスタノードスケジューラ (140-1、…、140-m) は、対応する計算機 (110-1、…、110-m) 上の各プロセッサ (120-11~120-1N1、…、120-m1~120-mNm) 内に存在する性能測定手段 (130-11~130-1N1、…、130-m1~130-mNm) を制御し、各プロセス (170-11~170-mLm) 実行時におけるプロセッサ (120-11~120-1N1、…、120-m1~120-mNm) のプロセッサ動作特性を採取する。このプロセッサ動作特性をクラスタスケジューラ (150) との間で交換し、クラスタ全体で各プロセスのプロセッサ動作特性に基づくスケジューリングを可能とする。すなわち、クラスタノードスケジューラ (140-1、…、140-m) がクラスタスケジューラ (150) に自計算機 (110-1、…、110-m) 上のプロセス (170-11~170-1L1、…、170-m1~170-mLm) のプロセッサ動作特性を渡すことで、クラスタスケジューラ (150) は計算機クラスタシステム全体でのプロセススケジューリングが可能となる。逆に、クラスタスケジューラ (150) が計算機 (110-1、…、110-m) へのプロセス割り当て時に、当該プロセスを他の計算機 (110-1、…、110-m) で実行した際に採取したプロセッサ動作特性を渡すことで、クラスタノードスケジューラ (140-1、…、140-m) はこれに基づきプロセッサの割り当てを決定できる。

### 【0023】

本実施の形態例では、クラスタスケジューラ (150) はオペレーティングシステム1 (140-1) 上に存在するが、本発明はクラスタスケジューラ (150) が計算機クラスタシステム内のどの部分に存在するかに関わりなく実現可能である。なぜなら、クラスタスケジューラ (150) 及びクラスタノードスケジューラ (140-1~140-m) も一種のプロセスであり、計算機内又は計算

機間にまたがったプロセス間通信は公知の技術で実現されているためである。これにより、例えばクラスタスケジューラ（150）をスケジューラ専用の計算機で実行する方法、あるいは、クラスタスケジューラ（150）自体の機能を計算機クラスタシステム上の複数計算機（110-1～110-m）に分散させて実現する方法も可能である。

## 【0024】

本実施の形態例では、各計算機（110-1、…、110-m）のオペレーティングシステム（160-1、…、160-m）内のスケジューリング機能を担当するクラスタスケジューラ（150）及びクラスタノードスケジューラ（140-1、…、140-m）が、各プロセッサ内の性能測定手段（130-11～130-1N1、…、130-m1～130-mNm）を制御して各プロセス（170-11～170-1L1、…、170-m1～170-mLm）のプロセッサ動作特性を採取し、この特性に基づき各プロセス（170-11～170-mLm）を各プロセッサ（130-11～130-mNm）に割り当てることにより、プロセス（170-11～170-mLm）毎の動作特性に基づく動的負荷分散が可能となる。これにより、プロセス毎のプロセッサ動作特性を考慮せずにプロセッサ（130-11～130-mNm）を割り当てる従来方式と比較し、より良いプロセッサ割り当てを実現でき、計算機クラスタシステムの性能を向上することができる。

## 【0025】

以下、本実施の形態例における計算機システムの構成及び動作を詳しく説明する。

## 【0026】

図2～図6は、本実施の形態例における計算機システムの構成、クラスタスケジューラ及びクラスタノードスケジューラの保持する情報を示す。

## 【0027】

図2は、本実施の形態例に係る計算機クラスタシステムのハードウェア構成を示す。

## 【0028】

図2の計算機クラスタシステムは、計算機1～3がネットワーク（200）により接続された構成を採る。

#### 【0029】

計算機1は、各々1MBのキャッシュメモリ（280-11、280-12）を持つ2つのプロセッサ（220-11、220-12）、メモリ（228-1）、ディスク（226-1）をシステム制御回路（224-1）により結合した構成を採る。プロセッサ（220-11、220-12）は、プロセッサバス（222-1）を共有する。計算機2は、各々2MBのキャッシュメモリ（280-21、280-12）を持つ2つのプロセッサ（220-21、220-22）、メモリ（228-2）、ディスク（226-2）をシステム制御回路（224-2）により結合した構成を採る。プロセッサ（220-21、220-22）は、プロセッサバス（222-2）を共有する。計算機3は、各々1MBのキャッシュメモリ（280-31、…、280-34）を持つ4つのプロセッサ（220-31、…、220-34）、メモリ（228-3）、ディスク（228-3）をシステム制御回路（224-3、224-31、224-32）により結合した構成を採る。2つのプロセッサ（220-31、220-32）はプロセッサバス（222-31）を介してシステム制御回路（224-31）に接続され、2つのプロセッサ（220-33、220-34）はプロセッサバス（222-32）を介してシステム制御回路（224-32）に接続される。

#### 【0030】

各プロセッサ（220-11～220-34）は、当該プロセッサの動作特性を採取可能な性能測定手段（230-11～230-34）を有する。また、計算機1上で動作するオペレーティングシステム（260-1～260-3）はクラスタスケジューラ（250）を、計算機1～3上で動作する各オペレーティングシステム（260-1～260-3）は前述のクラスタノードスケジューラ（250）を有する。

#### 【0031】

ここで、性能測定手段（230-11～230-34）を各プロセッサ（220-11～220-34）上に設ける構成に代えて、システム制御回路（224



ー 1 ～ 2 2 4 - 3 2) 内に設ける構成も可能である。この場合においても、プロセッサ (2 2 0 - 1 1 ～ 2 2 0 - 3 4) が対応するプロセッサバス (2 2 2 - 1 ～ 2 2 2 - 3 2) へ発行するメモリアクセスコマンド数等を測定でき、プロセススケジューリングに有用なシステム動作特性情報を得ることができる。従って、本発明はこの様な計算機についても適用可能である。

#### 【 0 0 3 2 】

図 3 は、クラスタスケジューラ (2 5 0) 及びクラスタノードスケジューラ (2 4 0 - 1 ～ 2 4 0 - 3) が保持するプロセッサ特性情報である。本実施の形態例において、プロセッサ特性情報はクラスタノード (計算機) 番号、計算機内のノード番号、プロセッサ番号で示されるプロセッサのキャッシュ容量、メモリアクセスレイテンシを保持する。本実施の形態例では、説明の便宜上プロセッサ (2 2 0 - 1 1 ～ 2 2 0 - 3 4) のコア部分の動作周波数、演算器の数等は全て同じとしておりプロセッサ特性情報に記載していないが、プロセッサ特性情報にこれらを含めて記載することも可能である。この場合、プロセッサ (2 2 0 - 1 1 ～ 2 2 0 - 3 4) のコア部分の差異を考慮したプロセススケジューリングが可能となるが、これについては以後適宜補足する。

#### 【 0 0 3 3 】

図 4 は、クラスタスケジューラ (2 5 0) 及びクラスタノードスケジューラ (2 4 0 - 1 ～ 2 4 0 - 3) が保持するノード特性情報である。本実施の形態例において、ノード特性情報は、クラスタノード (計算機) 番号、計算機内のノード番号で示されるノードのメモリアクセススループットを保持する。例えば、(クラスタノード番号、ノード番号) = (3, 1) のノード、すなわち図 2 におけるシステム制御回路 (2 2 4 - 3 1) を中心として構成されるノードは、メモリアクセススループット 0. 5 G B / s であることを示す。

#### 【 0 0 3 4 】

図 5 は、クラスタスケジューラ (2 5 0) 及びクラスタノードスケジューラ (2 4 0 - 1 ～ 2 4 0 - 3) が保持するクラスタノード特性情報である。本実施の形態例において、クラスタノード特性情報はクラスタノード (計算機) 番号で示されるクラスタノードのメモリアクセススループットを保持する。

## 【0035】

以上、図3～図5に示した特性情報は、計算機クラスタシステムの稼動中クラスタスケジューラ(250)及びクラスタノードスケジューラ(240-1～240-3)が保持する情報である。図3～図5の情報は、一つの方法としてディスク上のファイルシステムに保存しておき、計算機クラスタシステムの立ち上げ時にオペレーティングシステム(260-1～260-3)が当該ファイルを読み出してクラスタスケジューラ(250)及びクラスタノードスケジューラ(240-1～240-3)に引き渡す方法を採用することが可能である。別の方法としては、クラスタノードスケジューラ(240-1～240-3)が、計算機クラスタシステムの立ち上げ時若しくは適当なタイミングで図3～図5の特性情報を計測するベンチマークテストを実施する方法を採用することが可能である。このようなベンチマークテストとしては、プロセッサの性能特性に関してSPEC CPUベンチマーク(<http://www.spec.org>)、メモリアクセスループットに関してSTREAMベンチマーク(<http://www.cs.virginia.edu/stream>)、メモリアクセスレイテンシに関してlmbench(<http://reality.sgi.com/lm/lmbench>)等公知の技術を適用できる。

## 【0036】

図6は、クラスタスケジューラが保持するプロセス割り当て情報である。図6は、現在プロセスAP1、AP2が各々計算機1のプロセッサ(1, 1, 1)[図2中のプロセッサ220-11]、(1, 1, 2)[図2中のプロセッサ220-12]、プロセスAP3、AP4が各々計算機2のプロセッサ(2, 1, 1)[図2中のプロセッサ220-21]、(2, 1, 2)[図2中のプロセッサ220-22]、プロセスAP5～AP8が各々計算機3のプロセッサ(3, 1, 1)[図2中のプロセッサ220-31]、(3, 1, 2)[図2中のプロセッサ220-32]、(3, 2, 3)[図2中のプロセッサ220-33]、(3, 2, 4)[図2中のプロセッサ220-34]に割り当てられていることを示す。

## 【0037】

各クラスタノードスケジューラ(240-1～240-3)は、図6の情報の

うち、自計算機上で動作中のプロセスに関する情報のみを保持する。すなわち、計算機1は図6中AP1、AP2の行、計算機2は図6中AP3、AP4の行、計算機3は図6中AP5～AP8のプロセス割り当て情報を保持する。計算機内の各プロセッサ(220-11～220-34)にプロセスを割り当てる操作は、前述の様に各クラスタノードスケジューラ(240-1～240-3)により行われる。クラスタノードスケジューラ(240-1～240-3)は、当該計算機内のプロセッサ(220-11～220-34)に対するプロセス割り当ての変更時に新しい割り当てをクラスタスケジューラ(250)に通知し、クラスタノードスケジューラ(240-1～240-3)の保持するプロセス割り当て情報とクラスタスケジューラ(250)の保持するプロセス割り当て情報を一致させる。

## 【0038】

プロセス割り当て情報にはプロセス毎に対応して、図1に示した性能測定手段(230-11～230-34)により計測したプロセッサ動作特性を登録可能である。(図6は、計算機クラスタシステム上で初めてプロセスを割り当てた状態を示しており、プロセッサ動作特性が登録されていない様子を示す。)

図7～図12は、プロセススケジューリング方法とその動作を示す。

## 【0039】

図7は、図2の計算機クラスタシステム上に存在する3種類のプロセッサで、プロセスを動作させる場合の性能推測方法を示す。図7の性能推測方法は、キャッシュ1MB、メモリアクセスレイテンシ200nsのプロセッサ(220-11、220-12)を基準とした時に、他のプロセッサにおけるメモリアクセス待ち時間比率、メモリアクセススループットの性能予測値を導出する方法を示す。

## 【0040】

キャッシュ1MB、レイテンシ200nsのプロセッサ(220-11～220-12)におけるプロセス処理時間を1、メモリアクセス待ち時間比率を $M_w$ とする。この時、メモリアクセス待ちを除いたプロセッサの処理時間は $(1 - M_w)$ に相当する。

## 【0041】

まず、キャッシュ2MB、メモリアクセスレイテンシ200nsのプロセッサ(220-21~220-22)で同じプロセスを実行した場合の性能予測値を考える。キャッシュ容量が1MBから2MBに増加することで、キャッシュヒット率が向上してメモリアクセス回数が減少する結果、メモリアクセス待ち時間が低減する。また、メモリアクセス回数の減少に伴い、プロセスの要求するメモリアクセススループットも低減する。本実施の形態例では、このメモリアクセス待ち時間及びメモリアクセススループットの比を同一値 $E_{2M} (=2/3)$ としている。

## 【0042】

以上の結果、キャッシュ2MB、レイテンシ200nsの場合、プロセッサの処理時間は $(1-Mw)$ 、メモリアクセス待ち時間比率は $Mw \times E_{2M}$ となる。従って、キャッシュ2MB、レイテンシ200nsのメモリアクセス待ち時間 $Mw'$ は、 $Mw \times E_{2M} / \{ (1-Mw) + Mw \times E_{2M} \}$ となる。また、キャッシュ2MB、レイテンシ200nsのメモリアクセススループット $T'$ は、キャッシュ1MB、レイテンシ200ns時のメモリアクセススループット $T$ を用いて、 $T \times E_{2M} / \{ (1-Mw) + Mw \times E_{2M} \}$ となる。ここで、 $\{ \}$ の項による除算は、プロセスの処理時間短縮に伴い単位時間当たりのメモリアクセス量が増加することを反映している。

## 【0043】

同様に、キャッシュ1MB、メモリアクセスレイテンシ400nsのプロセッサ(220-31~220-34)では、メモリアクセスレイテンシが200nsから400nsに増加する結果のレイテンシ比 $E_{400ns} (=2)$ を用いて図7(2)に示す様に算出できる。ここで、メモリアクセススループット $T''$ 算出時の分子において、 $T$ に $E_{400ns}$ を乗じていないのは、キャッシュ容量が同一でありキャッシュヒット率が変わらないことによる。

## 【0044】

この性能推測方法を用いることで、図2の計算機クラスタシステム上のいずれかのプロセッサで採取したプロセッサ動作特性に基づき、他のプロセッサでの処

理性能を推定できる。

【 0 0 4 5 】

なお、プロセッサのコア部分の周波数、演算器の数等が異なる場合、プロセッサ処理時間を増減する係数を用いれば以上と同様に性能を推測できる。

【 0 0 4 6 】

本実施の形態例におけるプロセススケジューリング動作の概略は以下の通りである。

(1) 各プロセッサ (2 2 0 - 1 1 ~ 2 2 0 - 1 2、…、2 2 0 - 3 1 ~ 2 2 0 - 3 4) は、割り付けられたプロセスを実行する。この時、クラスタノードスケジューラ (2 4 0 - 1、…、1 4 0 - 3) は、自計算機内の各プロセッサ (2 2 0 - 1 1 ~ 2 2 0 - 1 2、…、2 2 0 - 3 1 ~ 2 2 0 - 3 4) が有する性能測定手段 (2 3 0 - 1 1 ~ 2 3 0 - 1 2、…、2 3 0 - 3 1 ~ 2 3 0 - 3 4) を制御し各プロセスのプロセッサ動作特性を計測する。

(2) 各クラスタノードスケジューラ (2 4 0 - 1、…、2 4 0 - 3) は、(1) で計測したプロセッサ動作特性をクラスタスケジューラ (2 5 0) に送付する。

(3) クラスタスケジューラ (2 5 0) は、各プロセスのプロセッサ動作特性に基づき、各プロセスを割り付けるプロセッサ (2 2 0 - 1 1 ~ 2 2 0 - 3 4) を選択する。

(4) (1) に戻る。

【 0 0 4 7 】

以下、上記 (1) ~ (4) について図 8 ~ 図 1 2 を用いて詳細を示す。

(1) プロセッサ動作特性の計測

クラスタスケジューラ (2 5 0) 及びクラスタノードスケジューラ (2 4 0 - 1 ~ 2 4 0 - 3) は、図 6 に従い各プロセスをプロセッサ (2 2 0 - 1 1 ~ 2 2 0 - 3 4) に割り付ける。この動作においては、クラスタスケジューラ (2 5 0) は各クラスタノードスケジューラ (2 4 0 - 1 ~ 2 4 0 - 3) の各々に当該計算機内で実行するプロセスに関するプロセス割り当て情報 (図 6) を送付する。各クラスタノードスケジューラ (2 4 0 - 1 ~ 2 4 0 - 3) は、クラスタスケジ

ューラ（２５０）から受け取ったプロセス割り当て情報を基に、各プロセスを当該計算機内の各プロセッサ（２２０－１１～２２０－３４）に割り付ける。

【００４８】

クラスタノードスケジューラ（２４０－１～２４０－３）は、プロセスをプロセッサ（２２０－１１～２２０－３４）に割り付ける直前に、当該プロセッサの性能測定手段（２３０－１１～２３０－３４）を制御しプロセッサ動作特性測定を開始する。そしてオペレーティングシステム（２６０－１～２６０－３）が規定するタイムスライス間隔の後、当該性能測定手段（２３０－１１～２３０－３４）を制御しプロセッサ動作特性測定を停止してプロセッサ動作特性を採取する。

（２） プロセッサ動作特性をクラスタスケジューラ（２５０）に送付

各クラスタノードスケジューラ（２４０－１～２４０－３）は、（１）で採取したプロセッサ動作特性情報をクラスタスケジューラ（２５０）に送付する。クラスタスケジューラ（２５０）は、これらを受けてプロセス割り当て情報の各プロセスのエントリにプロセッサ動作特性を付加する。図８は、図６に示したプロセス割り当て情報に基づきプロセッサ割り当てを行った結果、各プロセスのプロセッサ動作特性を採取した状態を示す。各プロセスに対応するプロセッサ動作特性は、プロセッサ番号、メモリアクセス待ち時間比率（図中メモリ待ち比率）、プロセスのメモリアクセス量（図中スループット）により示される。

（３） プロセスをプロセッサに割付

クラスタスケジューラ（２５０）は、図８に示したプロセス毎のプロセッサ動作特性に基づき新しいプロセッサ割り当てを決定する。

【００４９】

図９は、図８に示したプロセス毎のプロセッサ動作特性に基づき、図７の性能推定方法を使用して各プロセスを各プロセッサで動作させた際のプロセッサ動作特性を推測したものである。

【００５０】

図１０は、本実施の形態例におけるプロセススケジューリング方法を示す。

【００５１】

プロセスの処理時間は、図6の説明で示した様にプロセッサ処理時間とメモリアクセス待ち時間の合計となる。従って、プロセスの処理時間を短縮し処理性能を向上するには、メモリアクセス待ち時間比率を最小化する必要がある。本実施の形態例では、以下に示す3種類の方法を用いてメモリアクセス待ち時間比率を最小化する。

(i) 大容量キャッシュを持つプロセッサを使用

大容量のキャッシュを持つプロセッサでプロセスを実行することにより、キャッシュヒット率が向上し、メモリアクセスレイテンシの隠蔽効果、メモリアクセス量の低減効果の両方が得られる。メモリアクセスレイテンシの隠蔽効果により、プロセッサのメモリアクセス待ち時間が低減できる。また、メモリアクセス量の低減により、プロセッサ、ノード、クラスタノードの性能を超えたアクセス要求が発行されてメモリアクセス待ち時間が増大するのを防ぐ。

(ii) メモリアクセスレイテンシの小さいプロセッサを使用

メモリアクセスレイテンシが小さいプロセッサでプロセスを実行することで、メモリアクセスレイテンシが低減する。この結果、プロセッサのメモリアクセス待ち時間が低減できる。

(iii) プロセッサ/ノード/クラスタノード当たりのメモリアクセススループットが高いプロセッサ/計算機を使用

プロセッサ、ノード、クラスタノードのいずれかの性能を超えたアクセス要求が発行されると、当該部位で生じる待ちによりメモリアクセス待ち時間が増大する。この場合、プロセスをメモリアクセススループットが高いプロセッサ/計算機で実行することで、メモリアクセス待ち時間が低減できる。

【0052】

図9のプロセッサ動作特性推測値を元に、図10のプロセススケジューリング方法に従って、以下の様にメモリアクセス待ち時間比率を最小化する。

(1) クラスタスケジューラ(250)は、メモリアクセス待ち時間比率の低減能力の高いプロセッサから順にプロセスを割り当てる。本実施の形態例では、メモリアクセス待ち時間比率の低減能力の高いプロセッサ順は、キャッシュ2MB、メモリアクセスレイテンシ200nsのプロセッサ(220-21、220

-22)、キャッシュ1MB、メモリアクセスレイテンシ200nsのプロセッサ(220-11、220-12)、キャッシュ1MB、メモリアクセスレイテンシ400nsのプロセッサ(220-31~220-34)である。

(2) キャッシュ2MB、メモリアクセスレイテンシ200nsのプロセッサ(220-21、220-22)の割り当て時、クラスタスケジューラ(250)はこれらのプロセッサにおける各プロセスAP1~AP8のプロセッサ動作特性推測値及び実測値(図9)を比較し、メモリアクセス待ち時間比率の最も高いAP3、AP5を選択する。この時、メモリアクセス待ち時間比率に加えて、当該計算機に割り当てる全プロセスのメモリアクセス量の推測値及び実測値が、図5のクラスタノード特性情報で示される当該計算機のメモリアクセススループットと比較しなるべく超過しない様に選択する。

(3) キャッシュ1MB、メモリアクセスレイテンシ200nsのプロセッサ(220-11、220-12)の割り当て時、クラスタスケジューラ(150)はこれらのプロセッサにおけるAP3、AP5を除く各プロセスのプロセッサ動作特性推測値及び実測値(図9)を比較し、メモリアクセス待ち時間比率の最も高いAP6、AP8を選択する。この時、メモリアクセス待ち時間比率に加えて、当該計算機に割り当てる全プロセスのメモリアクセス量の推測値及び実測値が、図5のクラスタノード特性情報で示される当該計算機のメモリアクセススループットと比較しなるべく超過しない様に選択する。

(4) キャッシュ1MB、メモリアクセスレイテンシ400nsのプロセッサ(220-31~220-34)には、(2)(3)で選択済みのプロセスを除くAP1、AP2、AP4、AP7を選択する。この時、メモリアクセス待ち時間比率に加えて、当該計算機に割り当てる全プロセスのメモリアクセス量の推測値及び実測値が、図5のクラスタノード特性情報で示される当該計算機のメモリアクセススループットと比較しなるべく超過しない様に選択する。

(5) クラスタスケジューラ(250)は、(2)~(4)の選択に基づき各計算機のクラスタノードスケジューラ(240-1~240-3)にプロセスを割り付ける。

(6) 各クラスタノードスケジューラ(240-1~240-3)は、(5)



でクラスタスケジューラ(250)から割り当てられた各プロセスを自計算機内の各プロセッサ(220-11~220-34)に割り付ける。複数ノードを持つ計算機3では、各プロセッサ(220-31~220-34)へのプロセス割り付け時に図4に示したノード当たりのメモリアクセス性能を考慮する。すなわち、プロセスAP1、AP2、AP4、AP7を各プロセッサ(220-31~220-34)の割り付けにおいて、ノード当たりのメモリアクセススループットが0.5GB/sであることを考慮し、AP1、AP2を異なるノードに配置する。

## 【0053】

本実施の形態例では、説明の便宜上各プロセスのプロセッサ動作特性のみに基づきプロセススケジューリングを行っている。実際のオペレーティングシステムでは、各プロセスに実行待ち時間等を考慮した優先度を与え、優先度の高いプロセスを選択して実行する。本発明は、各プロセッサへのプロセス割り付け優先度を、各プロセスのプロセッサ動作特性に基づき加減することで、既存のプロセススケジューリングアルゴリズムに容易に組み込むことができる。

## 【0054】

図11に上記(1)~(6)に示したプロセススケジューリング動作により、図9の各プロセスのプロセッサ割り当てを再実行した結果を示す。この作業により、各プロセスのプロセッサ動作特性が図7に示した性能推測方法通りであった場合、プロセス処理性能はキャッシュ1MB、メモリアクセスレイテンシ200nsのプロセッサ1個に対して、7.53倍から7.99倍に向上する。

## 【0055】

図12に、新しいプロセッサ割り当てに基づくプロセス実行時のプロセッサ動作特性を測定し、測定結果を図8のプロセス割り当て情報に追加した状態を示す。この様に、プロセス割り当て情報には、処理が進むにつれ各プロセスを異なるプロセッサで動作させた際のプロセッサ動作特性が登録されていく。これにより、以後このプロセッサ動作特性の実測結果に基づきプロセススケジューリングが行える。

## 【0056】

また、プロセスの終了時又は計算機クラスタシステムのシャットダウン時に、プロセス割り当て情報に登録されたプロセッサ動作特性をファイルシステムに保存しておき、プロセスの実行時にファイルシステムに保存したプロセッサ動作特性を読み出すことで、プロセス実行開始時に以前の実行結果に基づき好適なプロセススケジューリングが可能となる。

## 【0057】

さらに、計算機クラスタシステム上の一部のプロセッサが性能測定手段（230-11～230-34）を持たない場合においても、クラスタスケジューラ（250）、クラスタノードスケジューラ（240-1～240-3）は、性能測定手段（230-11～230-34）を有するプロセッサ上でプロセスを実行した際に採取したプロセッサ動作特性を元に、これらのプロセッサのプロセス処理性能を推定できる。そして、クラスタスケジューラ（250）、クラスタノードスケジューラ（240-1～240-3）は、この推定を元に好適なプロセススケジューリングが可能となる。

## 【0058】

以上が本発明の実施の形態例1である。

## 【0059】

1以上の計算機を有する計算機クラスタシステムにおいて、各計算機のオペレーティングシステム（260-1、…、260-3）内のスケジューリング機能を担当するクラスタスケジューラ（250）及びクラスタノードスケジューラ（240-1、…、240-3）が、各プロセッサ（220-11～220-12、…、220-31～220-34）内の性能測定手段（230-11～230-12、…、230-31～230-34）を制御して各プロセス（170-11～170-1L1、…、170-m1～170-mLm）毎のプロセッサ動作特性を採取し、この特性に基づき各プロセス（170-11～170-mLm）を各プロセッサ（220-11～220-34）に割り当てることにより、プロセス（170-11～170-mLm）毎の動作特性に基づく動的負荷分散が可能となる。これにより、プロセス（170-11～170-mLm）毎のプロセッサ動作特性を考慮せずにプロセッサ（220-11～220-34）を割り当

てる従来方式と比較し、より良いプロセッサ割り当てを実現でき、計算機クラスタシステムの性能を向上できる。

## 《実施の形態例 2》

本発明の実施の形態例 2 を説明する。

### 【0060】

実施の形態例 2 は、実施の形態例 1 の変形であるため、相違点のみ図 1 3 ～ 図 1 4 を用いて説明する。

### 【0061】

本実施の形態例は、クラスタノードスケジューラ (240-1 ～ 240-3) が、性能測定手段 (230-11 ～ 230-34) を制御してメモリアクセス量の変化を採取し、これを用いて各プロセス毎にプロセッサの処理時間 (タイムスライス) を割り当てる際にタイムスライスの開始時刻及びタイムスライスの長さを最適化する点が異なる。

### 【0062】

本実施の形態例では、実施の形態例 1 の計算機 2 において、図 1 2 で示したプロセッサ動作特性を持つプロセス AP3、AP5 と、各々これらと同じプロセッサ動作特性を持つプロセス AP3'、AP5' の 4 つのプロセスを実行する際のタイムスライス最適化を示す。

### 【0063】

計算機 2 において、プロセッサ (220-21) 上で AP3 と AP3' を交互に、プロセッサ (220-22) 上で AP5 と AP5' を交互に実行する。図 1 2 で示した様に、プロセス AP3 (及び AP3') はメモリアクセス量 0.43 GB/s、AP5 (及び AP5') はメモリアクセス量 0.5 GB/s を持つ。このメモリアクセス量は、実際にはオペレーティングシステム (260-2) がこれらのプロセスにプロセッサ (220-21、220-22) を割り当てたタイムスライス内での平均値である。

### 【0064】

図 1 3 に、計算機 2 上の 2 プロセッサ (220-21 ～ 220-22) が、タイムスライス 10 ms で同時に AP3 と AP3'、AP5 と AP5' を切り替え

た場合のメモリアクセス量の変化を示す。AP3及びAP3'は、平均メモリアクセス量0.43GB/sであるが、最大0.9GB/sから最小0.2GB/sの範囲でメモリアクセス量が増減する。AP5及びAP5'は、平均メモリアクセス量0.5GB/sであるが、最大0.9GB/sから最小0.1GB/sの範囲でメモリアクセス量が増減する。メモリアクセス量は、プロセスをプロセッサ(220-21~220-22)に割り当てた直後が高く、プロセス割り当て後時間が経過すれば低下する。このような傾向は、プロセスをプロセッサ(220-21~220-22)に割り当てた当初は、当該プロセッサ内のキャッシュ(280-21~280-22)に当該プロセスの使用データが存在しないが、処理が進むにつれキャッシュ(280-21~280-22)に当該プロセスの使用データが登録されていくことによる。そして、他のプロセスが動作中は、当該プロセスにより登録されたデータがキャッシュ(280-21~280-22)から徐々に追い出される。図13では、AP3とAP3'、又は、AP5とAP5'を交互に実行した場合、プロセス切替直後はメモリアクセス量が大きい、プロセス切替後5ms以降はメモリアクセス量が低下する。

## 【0065】

プロセッサ(220-21)とプロセッサ(220-22)でのプロセス切替を同時に実行すると、AP3及びAP3'とAP5及びAP5'のメモリアクセス量最大の期間が重なり、図13に示した様に最大1.8GB/sものメモリアクセスが要求される。これに対し、計算機2の最大メモリアクセススループットは1.0GB/s(図4、図5)であり、これを超過する部分のプロセス処理性能は低下する。

## 【0066】

このプロセス処理性能低下を防ぐには、プロセッサ(220-21)とプロセッサ(220-22)のプロセス切替時刻をずらせ、AP3及びAP3'とAP5及びAP5'のメモリアクセス量が最大の期間をずらせば良い。図14は、AP3及びAP3'とAP5及びAP5'のメモリアクセス量が最大の期間をずらせたことにより、メモリアクセス要求量は最大1.1GB/sと計算機2の最大メモリアクセススループットにほぼ近い値まで低減できる。なお、プロセス切替

時刻をずらす量については、一つの例としてタイムスライス間隔 $S$ 、プロセス数 $P$ に対して $S/P$ ずつずらす方法が考えられる。また、プロセス切替時刻を少しずつずらして最大メモリアクセス量を算出し、この値がもっとも少ないずれ量を選択する方法も考えられる。

## 【0067】

また、プロセスの平均メモリアクセス量を低減するために、タイムスライスを長くする方法も考えられる。例えば、図13のプロセスAP3及びAP3'の場合、プロセス切替後5ms以降は当該プロセスが必要とするデータはほぼキャッシュ上に存在する。この結果、プロセス切替後5ms以降のメモリアクセス量は0.2GB/sとなる。従って、タイムスライスを20msに延長すれば、平均メモリアクセス量は0.32GB/s ( $= \{0.43\text{GB/s} \times 10\text{ms} + 0.2\text{GB/s} \times 10\text{ms}\} / 20\text{ms}$ )となる。この様に、平均メモリアクセス量の高いプロセスについては、タイムスライスの延長により平均メモリアクセス量を低減することができる。

以上が本発明の実施の形態例2である。

## 【0068】

本実施の形態例では、実施の形態例1において、クラスタノードスケジューラ(240-1~240-3)が、性能測定手段(230-11~230-34)を制御してメモリアクセス量の変化を採取し、これを用いて各プロセス毎にプロセッサのタイムスライスを割り当てる際にタイムスライスの開始時刻及びタイムスライスの長さを最適化する。これにより、各プロセスのメモリアクセス量が最大の期間が重なる結果、ノード又はクラスタノードの最大メモリアクセススループットを超えるメモリアクセス要求が発行されることによる性能低下を緩和できる。また、タイムスライスを延長することにより平均メモリアクセス量を低減でき、メモリアクセスの集中による性能低下を緩和できる。

## 《実施の形態例3》

本発明の実施の形態例3を図15を用いて説明する。

## 【0069】

実施の形態例3は、実施の形態例2における性能測定手段の構成を示すもので

あり、この部分に限って説明を行う。

【0070】

実施の形態例2では、性能測定手段(230-11~230-34)は、タイムスライス内でのメモリアクセス量の変化を採取する必要がある。このため、タイムスライスを10msとすると、10ms内に複数のサンプルポイントを設ける必要がある。本実施の形態例では、短いサンプル間隔で効率良く性能データを採取できる性能測定手段の構成を示す。

【0071】

図15は、本実施の形態例に係る性能測定手段の構成を示す。

【0072】

図15の性能測定手段は、プロセッサ又はシステム制御回路内に設けた性能測定回路(500)、メモリ空間(570)内に確保した性能測定制御用メモリ領域(580)、性能測定データ用メモリ領域(590)により構成される。

【0073】

性能測定回路(500)は、プロセッサ又はシステム制御回路内で発生する事象の数をカウントする性能測定データレジスタ(550)、プロセッサ又はシステム制御回路内でカウント可能な事象の中から性能測定データレジスタ(550)でカウントする事象を選択する性能測定制御レジスタ(530)、性能測定制御用メモリ領域(580)のベースアドレスを示すPMC\_BASEレジスタ(540)、性能測定データ用メモリ領域(590)のベースアドレスを示すPMD\_BASEレジスタ(560)、性能測定用メモリ領域内に格納可能な性能測定制御レジスタ(530)及び性能測定データレジスタ(550)の組の数を示すPM\_SIZEレジスタ(510)、性能測定用メモリ領域内で現在使用している性能測定制御レジスタ(530)及び性能測定データレジスタ(550)の組を示すPM\_OFFSETレジスタ(520)からなる。

【0074】

本実施の形態例においては、性能測定制御用メモリ領域(580)内のPM\_OFFSETレジスタ(520)で示される位置から性能測定用の設定を読みだして、性能測定制御レジスタ(530)に設定し、この設定により指定される事

象を性能測定データレジスタ（550）を用いてカウントし、あらかじめ設定した時間の後にカウントした値を性能測定データレジスタ（550）から性能測定データ用メモリ領域（590）内のPM\_OFFSETレジスタ（520）で示される位置へ保存し、その後PM\_OFFSETレジスタ（520）をインクリメントして次のカウントを行う。以上の処理を、オペレーティングシステム等の介在無しに実現することにより、性能測定手段をソフトウェアで制御するオーバーヘッドを抑え、短いサンプル間隔で効率良く性能データを採取できる性能測定手段を提供する。

## 【0075】

以下、性能測定回路（500）の動作を詳細に示す。

## （1） 性能測定制御レジスタ（530）の設定

性能測定制御用メモリ領域（580）から性能測定用の設定を読みだし、性能測定制御レジスタ（530）に設定する。

## 【0076】

本実施の形態例においては、性能測定制御レジスタ（530）は、8Bのレジスタ4個からなるものとする。この時、（1）で読み出す設定は、PMC\_BASEレジスタ値+PM\_OFFSETレジスタ値×32Bで示されるアドレスから始まる32Bのメモリ領域に格納されている。性能測定回路（500）は、このデータを読み出し性能測定制御レジスタ（530）に設定する。

## （2） 性能測定データレジスタ（550）の設定

本実施の形態例の性能測定手段のカウント動作は以下の2通りが存在する。

・性能測定データ用メモリ領域（580）から前回までにカウントした性能測定データ値を読み出し、性能測定データレジスタ（550）に設定する。本実施の形態例においては、性能測定データレジスタ（550）は、8Bのレジスタ4個からなるものとする。この時、読み出す性能測定データ値は、PMD\_BASEレジスタ値+PM\_OFFSETレジスタ値×32Bで示されるアドレスから始まる32Bのメモリ領域に格納されている。性能測定回路（500）は、このデータを読み出し性能測定データレジスタ（550）に設定する。

## 【0077】

これにより、比較的長い期間の間に、対応する性能測定制御レジスタ（530）に設定された事象が発生した回数をサンプルすることができる。例えば、60秒のプロセス動作時に400種類の事象（PM\_SIZE=100）を1msの切替間隔で採取すると、100msで全設定を一巡するため、各事象当たり600回のサンプルが可能である。この様にサンプル回数を増やすことで、数個の性能測定レジスタの組を用いて数百個の事象の各々の生起回数をほぼ正確に測定できる。このような測定方法は、D. Bhandarkar他の「Performance Characterization of the Pentium Pro Processor, In Proceedings of the Third International Symposium on High-Performance Computer Architecture, page 288-297, Feb. 1997」に示されている公知の技術である。この論文では、Pentium Proプロセッサの性能測定手段をソフトウェアで制御し、5秒間隔で性能測定レジスタの設定を切り替えて性能測定を行っている。本発明の実施の形態例における性能測定手段を使用すれば、ソフトウェアの介在無しに性能測定レジスタの設定を切り替え可能となり、切り替え間隔を大幅に短くできる。

・性能測定データレジスタ（550）を”0”に設定し、加算を開始する。これにより、当該期間内での事象の発生回数がカウントできる。

#### 【0078】

以上2通りの動作を、性能測定の目的に合わせて使い分けることができる。

#### （3） 性能測定

性能測定制御レジスタ（530）に設定された事象の発生回数を、性能測定データレジスタ（550）にカウントする。

#### （4） 性能測定データレジスタ（550）の値を性能測定データ用メモリ領域（590）に格納

あらかじめ設定された性能測定間隔の後、性能測定データレジスタ（550）の値を、性能測定データ用メモリ領域（590）の対応するアドレス（PMD\_BASEレジスタ値+PM\_OFFSETレジスタ値×32B）に格納する。



## (5) PM\_OFFSETレジスタ (520) のインクリメント

PM\_OFFSETレジスタ (520) をインクリメントし、性能測定用メモリ領域内の次の性能測定レジスタを指す様に移動する。この時、PM\_OFFSETレジスタ (520) がPM\_SIZEレジスタ (510) より大きくなった場合、PM\_OFFSETレジスタ (520) を” 0 ” とする。

【 0 0 7 9 】

以上が本発明の実施の形態例 3 である。

【 0 0 8 0 】

実施の形態例 3 においては、性能測定回路 (500) は性能測定制御用メモリ領域 (580) から性能測定用の設定を読みだして性能測定制御レジスタ (530) に設定し、この設定により指定される事象を性能測定データレジスタ (550) を用いてカウントし、あらかじめ設定した時間の後に、カウントした値を性能測定データレジスタ (550) から性能測定データ用メモリ領域 (590) に保存する。性能測定回路 (500) は、この動作を性能測定用メモリ領域内の各エントリに対して順次切り替えながら行う。これにより性能測定手段をソフトウェアで制御するオーバーヘッドを抑え、短いサンプル間隔で効率良く性能データを採取できる性能測定手段を提供する。

【 0 0 8 1 】

## 【発明の効果】

本発明によれば、1 以上の計算機を有する計算機クラスタシステムにおいて、各計算機のオペレーティングシステム内のスケジューリング機能を担当するクラスタスケジューラ及びクラスタノードスケジューラが、各プロセッサ又はシステム制御回路内の性能測定手段を制御して各プロセス毎のプロセッサ動作特性を採取し、この特性に基づき各プロセスを各プロセッサに割り当てることにより、プロセス毎の動作特性に基づく動的負荷分散が可能となる。これにより、プロセス毎のプロセッサ動作特性を考慮せずにプロセッサを割り当てる従来方式と比較し、より良いプロセッサ割り当てを実現でき、計算機クラスタシステムの性能を向上できる。

【 0 0 8 2 】

また、クラスタノードスケジューラが、性能測定手段を制御してメモリアクセス量の変化を採取し、これを用いて各プロセス毎にプロセッサのタイムスライスを割り当てる際にタイムスライスの開始時刻及びタイムスライスの長さを最適化する。これにより、各プロセスのメモリアクセス量が最大の期間が重なる結果、ノード又はクラスタノードの最大メモリアクセススループットを超えるメモリアクセス要求が発行されることによる性能低下を緩和できる。また、タイムスライスを延長することにより平均メモリアクセス量を低減でき、メモリアクセスの集中による性能低下を緩和できる。

#### 【 0 0 8 3 】

さらに、性能測定回路は性能測定制御用メモリ領域から性能測定用の設定を読みだして性能測定制御レジスタに設定し、この設定により指定される事象を性能測定データレジスタを用いてカウントし、あらかじめ設定した時間の後に、カウントした値を性能測定データレジスタから性能測定データ用メモリ領域に保存する。性能測定回路は、この動作を性能測定用メモリ領域内の各エントリに対して順次切り替えながら行う。これにより、性能測定手段をソフトウェアで制御するオーバーヘッドを抑え、短いサンプル間隔で効率良く性能データを採取できる。

#### 【図面の簡単な説明】

##### 【図 1】

本発明の実施の形態例 1 に係る計算機クラスタシステムの概略図。

##### 【図 2】

本発明の実施の形態例 1 に係る計算機クラスタシステムの構成図。

##### 【図 3】

本発明の実施の形態例 1 に係るプロセッサ特性情報を示す図。

##### 【図 4】

本発明の実施の形態例 1 に係るノード特性情報を示す図。

##### 【図 5】

本発明の実施の形態例 1 に係るクラスタノード特性情報を示す図。

##### 【図 6】

本発明の実施の形態例 1 に係るプロセス割り当て情報を示す図。

【図 7】

本発明の実施の形態例 1 に係る各プロセッサの性能推定方法を示す図。

【図 8】

本発明の実施の形態例 1 に係るプロセス割り当て情報を示す図。

【図 9】

本発明の実施の形態例 1 に係るプロセッサ動作特性推定値を示す図。

【図 1 0】

本発明の実施の形態例 1 に係るプロセススケジューリング方法を示す図。

【図 1 1】

本発明の実施の形態例 1 に係るプロセッサ動作特性推定値を示す図。

【図 1 2】

本発明の実施の形態例 1 に係るプロセス割り当て情報を示す図。

【図 1 3】

本発明の実施の形態例 2 に係るプロセス同時切替時のメモリアクセス量を示す図。

【図 1 4】

本発明の実施の形態例 2 に係るプロセス非同時切替時のメモリアクセス量を示す図。

【図 1 5】

本発明の実施の形態例 3 に係る性能測定手段を示す図。

【符号の説明】

1 0 0 … ネットワーク、

1 1 0 - 1 ~ 1 1 0 - m … 計算機、

1 2 0 - 1 1 ~ 1 2 0 - m N m … プロセッサ、

1 3 0 - 1 1 ~ 1 3 0 - m N m … 性能測定手段、

1 4 0 - 1 ~ 1 4 0 - m … クラスタノードスケジューラ、

1 5 0 … クラスタスケジューラ、

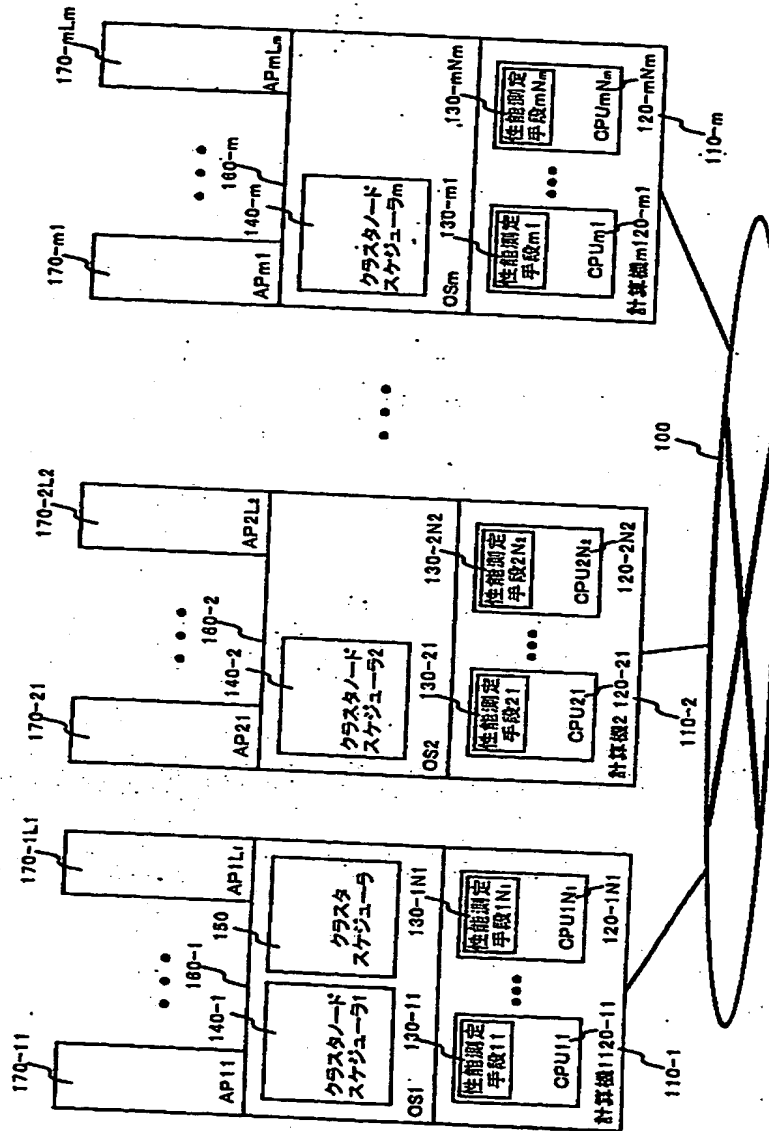
1 6 0 - 1 ~ 1 6 0 - m … オペレーティングシステム、

1 7 0 - 1 1 ~ 1 7 0 - m L m … プロセス、

200…ネットワーク、  
220-11～220-34…プロセッサ、  
222-1～222-32…プロセッサバス、  
224-1～224-32…システム制御回路、  
226-1～226-3…ディスク、  
228-1～228-3…メモリ、  
230-11～230-34…性能測定手段、  
240-1～240-3…クラスタノードスケジューラ、  
250…クラスタスケジューラ、  
260-1～260-3…オペレーティングシステム、  
280-11～280-34…キャッシュ、  
500…性能測定回路、  
510…PM\_SIZEレジスタ、  
520…PM\_OFFSETレジスタ、  
530…性能測定制御レジスタ、  
540…PMC\_BASEレジスタ、  
550…性能測定データレジスタ、  
560…PMD\_BASEレジスタ、  
570…メモリ空間、  
580…性能測定制御用メモリ領域、  
590…性能測定データ用メモリ領域。

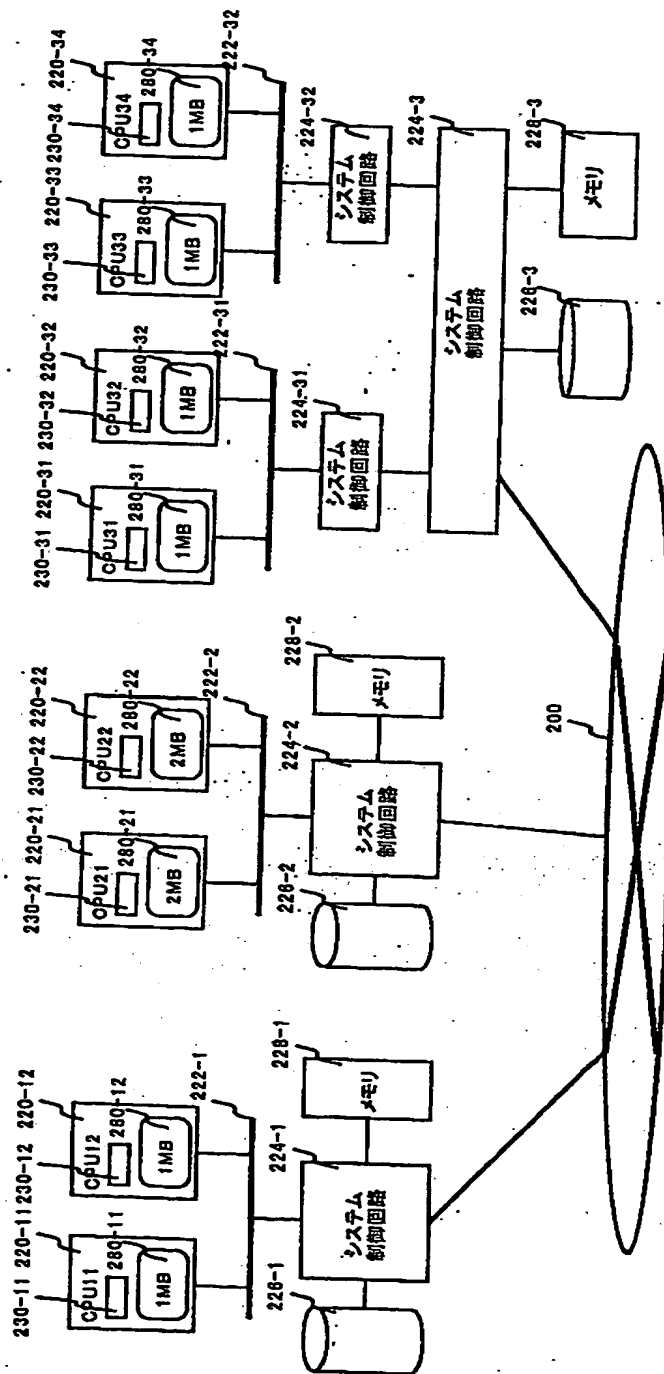
【書類名】 図面

【図 1】



【図 2】

図 2



【図3】

図3

クラスター ノード	ノード	CPU	キャッシュ 容量	メモリ レイテン
1	1	1	1MB	200ns
1	1	2	1MB	200ns
2	1	1	2MB	200ns
2	1	2	2MB	400ns
3	1	1	1MB	400ns
3	1	2	1MB	400ns
3	2	3	1MB	400ns
3	2	4	1MB	400ns

【図4】

図4

クラスターノード	ノード	ノードスループット
1	1	1GB/s
2	1	1GB/s
3	1	0.5GB/s
3	2	0.5GB/s

【図 5】

図 5

クラスタノード	クラスタノード スループット
1	1GB/s
2	1GB/s
3	1GB/s

【図 6】

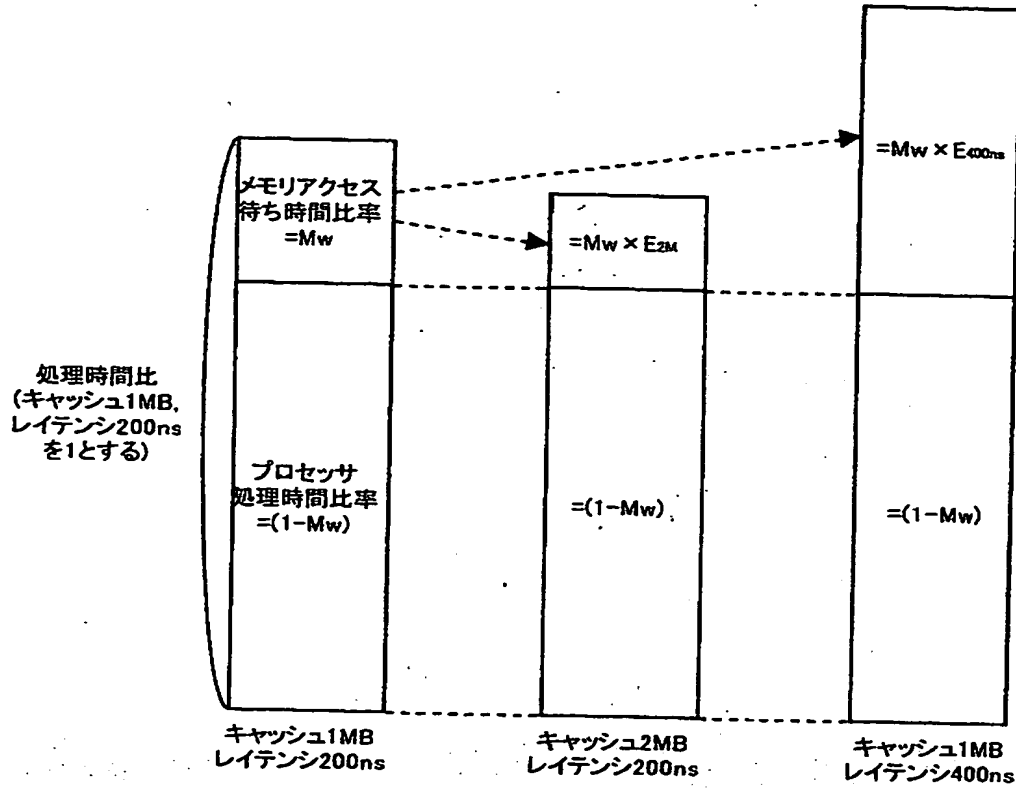
図 6

AP	CPU	プロセッサ 動作特性
AP1	(1,1,1)	→ none
AP2	(1,1,2)	→ none
AP3	(2,1,1)	→ none
AP4	(2,1,2)	→ none
AP5	(3,1,1)	→ none
AP6	(3,1,2)	→ none
AP7	(3,2,3)	→ none
AP8	(3,2,4)	→ none



【図7】

図7



- $M_w$  : キャッシュ1MB, レイテンシ200nsにおけるメモリアクセス待ち時間比率(%)
- $T$  : キャッシュ1MB, レイテンシ200nsにおけるメモリアクセス量(GB/s)
- $E_{2M}$  : キャッシュサイズを1MB→2MBとした時のレイテンシ/スループット比 ( $=2/3$ )
- $E_{400ns}$  : メモリアクセスレイテンシを200ns→400nsとした時のレイテンシ比 ( $=2$ )

【図 8】

図 8

AP	CPU	プロセッサ 動作特性		
AP1	(1,1,1)	CPU	メモリ待ち比率	スループット
		(1,1,1)	5%	0.42GB/s
AP2	(1,1,2)	CPU	メモリ待ち比率	スループット
		(1,1,2)	5%	0.42GB/s
AP3	(2,1,1)	CPU	メモリ待ち比率	スループット
		(2,1,1)	22%	0.44GB/s
AP4	(2,1,2)	CPU	メモリ待ち比率	スループット
		(2,1,2)	9%	0.10GB/s
AP5	(3,1,1)	CPU	メモリ待ち比率	スループット
		(3,1,1)	53%	0.51GB/s
AP6	(3,1,2)	CPU	メモリ待ち比率	スループット
		(3,1,2)	33%	0.25GB/s
AP7	(3,2,3)	CPU	メモリ待ち比率	スループット
		(3,2,3)	10%	0.10GB/s
AP8	(3,2,4)	CPU	メモリ待ち比率	スループット
		(3,2,4)	31%	0.30GB/s

【図9】

図9

プロセス名	プロセッサ割当	キャッシュ1MB,レイテンシ200nsのプロセッサ			キャッシュ2MB,レイテンシ200nsのプロセッサ			キャッシュ1MB,レイテンシ400nsのプロセッサ		
		Mw	T	性能比	Mw	T	性能比	Mw	T	性能比
AP1	(3,1,1)	5%	0.42GB/s	1.00	3%	0.28GB/s	1.02	10%	0.40GB/s	0.95
AP2	(3,2,4)	5%	0.42GB/s	1.00	3%	0.28GB/s	1.02	10%	0.40GB/s	0.95
AP3	(2,1,1)	30%	0.60GB/s	1.00	22%	0.44GB/s	1.11	46%	0.46GB/s	0.77
AP4	(3,1,2)	13%	0.15GB/s	1.00	9%	0.10GB/s	1.05	23%	0.13GB/s	0.88
AP5	(2,1,2)	36%	0.69GB/s	1.00	27%	0.52GB/s	1.14	53%	0.51GB/s	0.74
AP6	(1,1,1)	20%	0.30GB/s	1.00	14%	0.21GB/s	1.07	33%	0.25GB/s	0.83
AP7	(3,2,3)	5%	0.11GB/s	1.00	3%	0.07GB/s	1.02	10%	0.10GB/s	0.95
AP8	(1,1,2)	18%	0.35GB/s	1.00	13%	0.25GB/s	1.06	31%	0.30GB/s	0.85

性能比計=7.53

【図10】

図10

	メモリスループット要求量大	メモリスループット要求量小
メモリアクセス待ち時間比率大	① 大容量キャッシュのCPU使用 ② メモリレイテンシ小のCPU使用 ③ CPU/ノード/クラスターノード当りのメモリスループットが高いプロセッサ/計算機使用...	① 大容量キャッシュのCPU使用 ② メモリレイテンシ小のCPU使用
メモリアクセス待ち時間比率小	① 大容量キャッシュのCPU使用 ② CPU/ノード/クラスターノード当りのメモリスループットが高いプロセッサ/計算機使用	...

【図11】

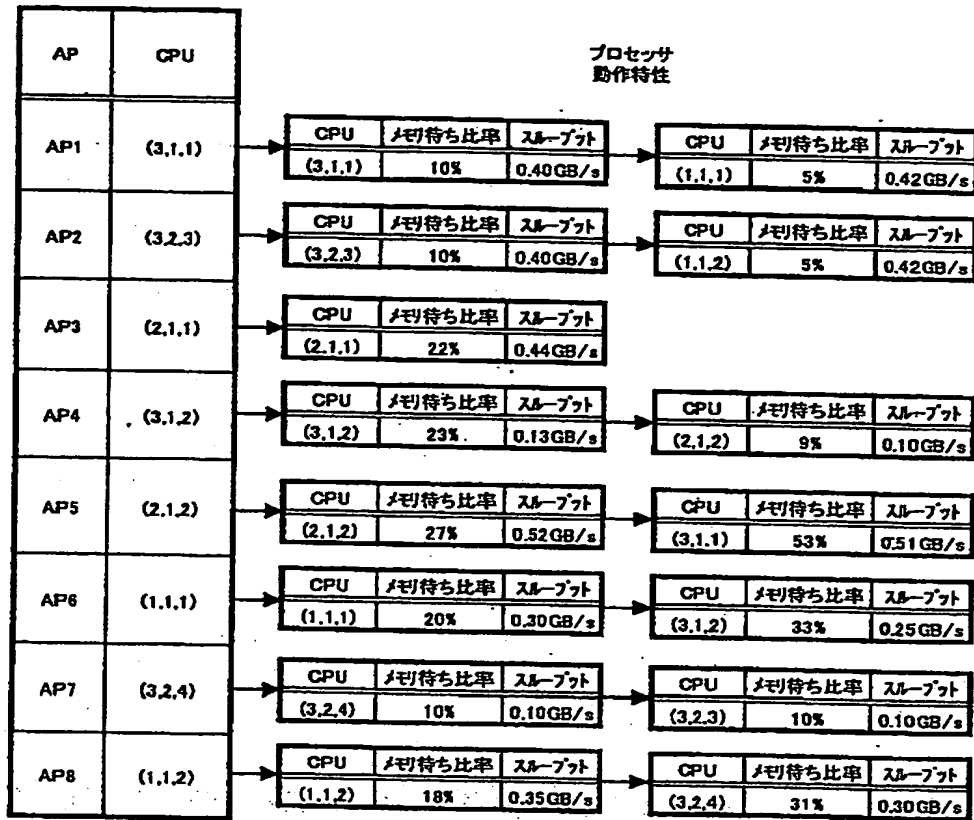
図11

プロセス名	プロセッサ割当	キャッシュ1MB,レイテンシ200nsのプロセッサ			キャッシュ2MB,レイテンシ200nsのプロセッサ			キャッシュ1MB,レイテンシ400nsのプロセッサ		
		Mw'	T	性能比	Mw'	T	性能比	Mw'	T	性能比
AP1	(3,1,1)	5%	0.42GB/s	1.00	3%	0.28GB/s	1.02	10%	0.40GB/s	0.95
AP2	(3,2,4)	5%	0.42GB/s	1.00	3%	0.28GB/s	1.02	10%	0.40GB/s	0.95
AP3	(2,1,1)	30%	0.60GB/s	1.00	22%	0.44GB/s	1.11	46%	0.46GB/s	0.77
AP4	(3,1,2)	13%	0.15GB/s	1.00	9%	0.10GB/s	1.05	23%	0.13GB/s	0.88
AP5	(2,1,2)	36%	0.69GB/s	1.00	27%	0.52GB/s	1.14	53%	0.51GB/s	0.74
AP6	(1,1,1)	20%	0.30GB/s	1.00	14%	0.21GB/s	1.07	33%	0.25GB/s	0.83
AP7	(3,2,3)	5%	0.11GB/s	1.00	3%	0.07GB/s	1.02	10%	0.10GB/s	0.95
AP8	(1,1,2)	18%	0.35GB/s	1.00	13%	0.25GB/s	1.06	31%	0.30GB/s	0.85

性能比計=7.99

【図 12】

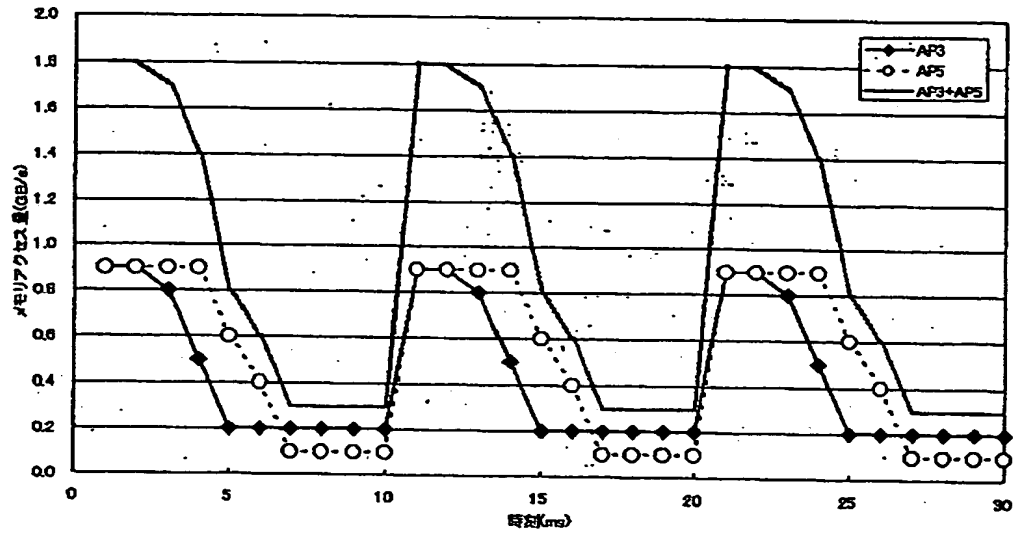
図 12



【図13】

図13

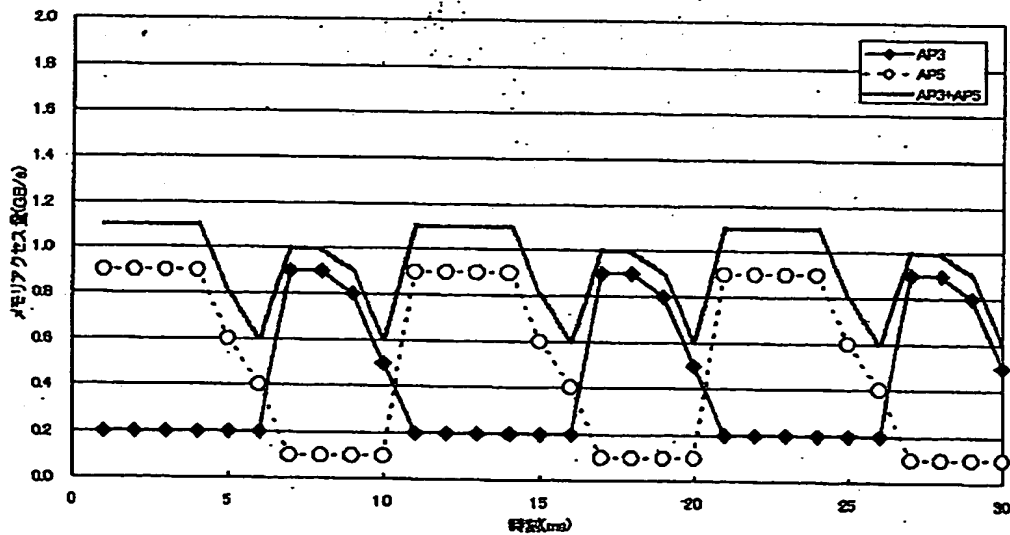
プロセス同時切替時のメモリアクセス量



【図14】

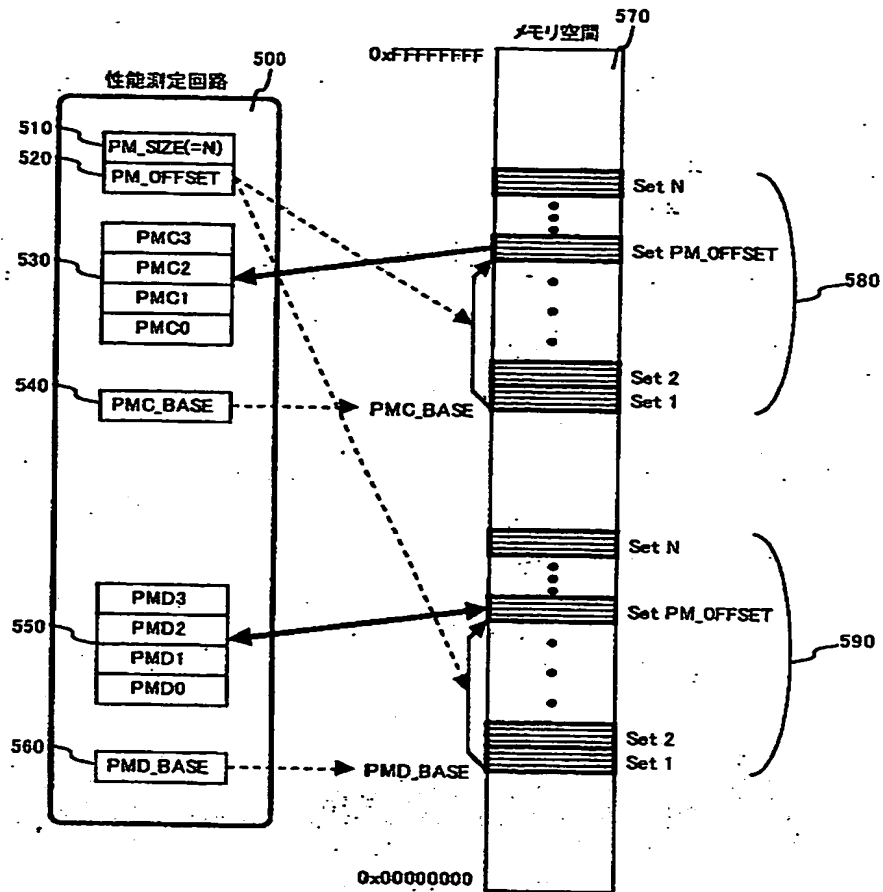
図14

プロセス非同期切替時のメモリアクセス量



【図15】

図15



【書類名】 要約書

【要約】

【課題】 複数プロセッサを有する計算機又は計算機クラスタシステムにおいて、プロセスの動作特性を実測し、プロセス動作特性実測値に基づきプロセススケジューリングを行うことで、処理性能を向上させる。

【解決手段】 計算機又は計算機クラスタシステムにおいて、各計算機で動作するオペレーティングシステム上にスケジューリング機能を設ける。前記スケジューリング機能は、各計算機内のプロセッサ又はシステム制御回路に設けた性能測定手段を制御し、各プロセスのプロセッサ動作特性を採取する。そして、前記スケジューリング機能は、このプロセッサ動作特性を元に各プロセッサ上で各々のプロセスを動作させた場合の動作特性を推測し、各プロセスのプロセッサへの割り当てを最適化する。

【選択図】 図 1



認定・付加情報

特許出願の番号	特願2001-192174
受付番号	50100923157
書類名	特許願
担当官	第三担当上席 0092
作成日	平成13年 6月27日

<認定情報・付加情報>

【提出日】 平成13年 6月26日

出 願 人 履 歴 情 報

識別番号 [000005108]

1. 変更年月日 1990年 8月31日

[変更理由] 新規登録

住 所 東京都千代田区神田駿河台4丁目6番地

氏 名 株式会社日立製作所